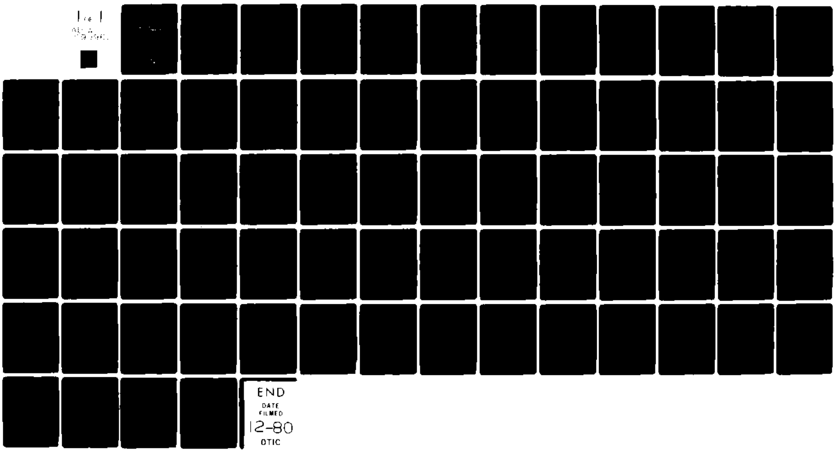
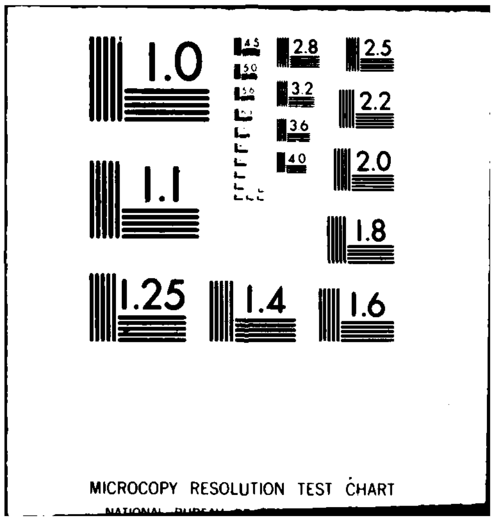
AD-A090 962   TRW DEFENSE AND SPACE SYSTEMS GROUP REDONDO BEACH CA   F/G 9/2
AIRBORNE SYSTEMS SOFTWARE ACQUISITION ENGINEERING GUIDEBOOK FOR--ETC(U)
FEB 80   T WINSLOW   F33657-76-C-0677
UNCLASSIFIED   TRW-30323-6018-TU-00   ASD-TR-80-5021   NL





MICROCOPY RESOLUTION TEST CHART

LEVEL II

ASD-TR-80-5021

1

Airborne Systems Software Acquisition Engineering Guidebook for MICROPROCESSORS AND FIRMWARE

AD A090962

FEBRUARY 1980

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

DTIC
ELECTE
OCT 3 0 1980
S D
E

PREPARED FOR
DEPUTY FOR ENGINEERING
AERONAUTICAL SYSTEMS DIVISION
WRIGHT-PATTERSON AFB, OH 45433

DDC FILE COPY

80 10 28 045

PREPARED BY
TRW DEFENSE AND SPACE SYSTEMS GROUP
ONE SPACE PARK
REDONDO BEACH, CA 90278

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

John M. Hoeferlin

JOHN M. HOEFERLIN, Project Engineer
Information Engineering Division

Richard J. Sylvester

RICHARD J. SYLVESTER
ASD Computer Resources Focal Point

FOR THE COMMANDER

Robert P. Lavoie

ROBERT P. LAVOIE, Colonel, USAF
Director of Avionics Engineering
Deputy for Engineering

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify ASD/ENAIA, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
2. GOVT ACCESSION NO. AD-A090 962
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
(6) Airborne Systems Software Acquisition Engineering Guidebook for Microprocessors and Firmware

5. TYPE OF REPORT & PERIOD COVERED
(9) Final rept.

6. PERFORMING ORG. REPORT NUMBER
(14) TRW-30323-6018-TU-00

7. AUTHOR(s)
(10) T. Winslow

8. CONTRACT OR GRANT NUMBER(s)
(15) F33657-76-C-0677

9. PERFORMING ORGANIZATION NAME AND ADDRESS
TRW Defense and Space Systems Group
One Space Park
Redondo Beach, Ca 90278 (11) Feb 80

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
PE64740F
(16) ~~Project~~ 2238

11. CONTROLLING OFFICE NAME AND ADDRESS
Hq ASD/ENAI
Wright-Patterson AFB, Ohio 45433

12. REPORT DATE
February 1980 (12) 72

13. NUMBER OF PAGES
74

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
(18) ASD (19) TR-80-5022

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release, Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Micro	PROM	EAROM	Guidebook	Digital Device
Microprocessor	RAM	EEROM	Digital	
Firmware	SSI	MSI	Programmable	
LSI	Small Scale Integration			

Large Scale Integration, Medium Scale Integration

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is one of a series of guidebooks whose purpose is to assist Air Force Program Office and Engineering personnel in the acquisition and engineering of airborne systems software. This guidebook provides guidance in planning the acquisition and support of microprocessor and microprocessor based firmware embedded in airborne systems. It provides definition of terminology and concepts along with guidance in selecting, planning, organizing, staffing, managing and controlling microprocessor and firmware embedded in systems. It provides detailed guidance for the acquisition,

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 409637
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 Continued.

management and support of microprocessors and firmware. It identifies the responsibilities of the participating organizations and discusses methods, products and problems.

Accession For	
NTIS GRA&I	☒
DDC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

PREFACE

This guidebook is one of a series of guidebooks intended to assist Air Force Program Office and Engineering personnel in software acquisition engineering for airborne systems. The contents of the guidebooks will be revised periodically to reflect changes in software acquisition policies and practices and feedback from users.

This guidebook was prepared under the direction of the Aeronautical Systems Division, Deputy for Engineering (ASD/EN) in coordination with the Space Division, Deputy for Acquisition Management (SD/AQM).

The entire series of Software Acquisition Engineering Guidebooks (Airborne Systems) is listed below along with ASD Technical Report numbers and NTIS accession numbers where available.

Regulations, Specifications and Standards	ASD-TR-78-6	ADA058428
Reviews and Audits	ASD-TR-78-7	ADA058429
Software Quality Assurance	ASD-TR-78-8	ADA059068
Configuration Management	ASD-TR-79-5024	ADA076542
Computer Program Documentation Requirements	ASD-TR-79-5025	ADA076543
Statements of Work and Requests for Proposal	ASD-TR-79-5026	ADA076544
Requirements Analysis and Specification	ASD-TR-79-5027	
Verification, Validation and Certification	ASD-TR-79-5028	
Microprocessors and Firmware	ASD-TR-80-5021	
Software Development Planning and Control	ASD-TR-80-5022	
Software Testing and Evaluation	ASD-TR-80-5023	
* Contracting for Software Acquisition	ASD-TR-80-5024	

* Software Cost Analysis and Estimating	ASD-TR-80-5025
* Supportable Airborne Software	ASD-TR-80-5026
* Software Development and Support Facilities	ASD-TR-80-5027
* SAE Guidebooks - Application and Use	ASD-TR-80-5028

---

* These Guidebooks Available Fall 1980.

CONTENTS

CONTENTS (Concluded)

TABLES

ILLUSTRATION

ABBREVIATIONS AND ACRONYMS

AGE	Aerospace Ground Equipment
AISF	Avionics Integration Support Facility
ATE	Automatic Test Equipment
CCD	Charge-Coupled Device
CI	Configuration Item
CMAC	Computer Monitor and Control
CMD	Configuration Management Data
CMP	Configuration Management Plan
CPDP	Computer Program Development Plan
CPIN	Computer Program Identification Number
CPU	Central Processing Unit
CRISP	Computer Resources Integrated Support Plan
CRT	Cathode Ray Tube
CRWG	Computer Resource Working Group
DID	Data Item Description
ECS	Embedded Computer System
EAROM	Electrically Alterable Read-Only Memory
EEROM	Electrically Eraseable Read-Only Memory
EIA	Electronic Industries Association
EPROM	Eraseable Programmable Read-Only Memory
FSED	Full Scale Engineering Development
HOL	High-Order Language
ICE	In-Circuit Emulation/Emulator

ABBREVIATIONS AND ACRONYMS (Concluded)

ILSP	Integrated Logistics Support Plan
I/O	Input/Output
IV&V	Independent Verification and Validation
LCC	Life Cycle Cost
LOL	Low-Order Language
LSI	Large-Scale Integration
MDS/MPS	Microcomputer Development/Prototyping System
MICRO	Microprocessor
MSI	Medium-Scale Integration
OPR	Office of Primary Responsibility
PLA	Programmable Logic Array
PMD	Program Management Directive
PMP	Program Management Plan
PMRT	Program Management Responsibility Transfer
PROM	Programmable Read-Only Memory
RAM	Random-Access Memory
ROM	Read-Only Memory
SSI	Small-Scale Integration
T&E	Test and Evaluation
UUT	Unit Under Test

1. INTRODUCTION

Microprocessors and firmware have a large role to play in airborne systems. Besides the standard advantages of digital technology, such as flexibility, reliability, stability, repeatability, arbitrary precision, ease of construction of complex transfer functions, time-sharing, and re-usability, Large Scale Integrated (LSI) circuits have added advantages in size, power, economy and variety, resulting from the advance of mass-production techniques. A key role of firmware is to "localize" the flexibility in embedded computer systems - the functional performance of a large digital subsystem can then be dramatically and usefully changed merely by replacing a few firmware components. Other advantages are the eas of distribution of functionality in an embedded computer system by the use of multiple program memories, the elimination of a load/verify step, and data security (including nuclear hardness).

1.1 PURPOSE OF THIS GUIDEBOOK

The purpose of this guidebook is to provide guidance in planning the acquisition and support of microprocessors and microprocessor-based firmware embedded in Air Force airborne systems. It will not address the type of firmware associated with the digital design technique known as microprogramming, i.e., the emulation of computers, implementation of instruction sets, etc.

1.2 RELATIONSHIP TO OTHER GUIDEBOOKS

This guidebook is one of a series of interrelated volumes (the Software Acquisition Engineering Guidebooks) designed to assist Air Force Program Office and engineering personnel in effective software acquisition engineering for airborne systems. The guidelines, checklists, and references in this guidebook serve to supplement the guidance in the other guidebooks of this series in relation to the peculiar subject matter of this book.

1.3 CONTENTS OF THIS GUIDEBOOK

1.3.1 Section 1: Introduction

Provides orientation and describes the purpose, scope, and contents of this guidebook.

1.3.2 Section 2: Relevant Documents

Lists the government regulations, specifications, and standards relevant to microprocessors and firmware.

1.3.3 Section 3: General Guidelines for Microprocessors and Firmware

Provides definition of terminology and concepts along with general guidance to the Program Office and engineering personnel in selecting, planning, organizing, staffing, managing, and controlling microprocessors and firmware embedded in airborne systems.

1.3.4 Section 4: Specific Guidance for Microprocessors and Firmware

Provides detailed guidance for the acquisition and management of microprocessors and firmware. Identifies the responsibilities of the participating organizations and discusses applicable concepts, methods, products, and problems.

1.3.5 Appendix A: Glossary of Key Terms

Provides short definitions of the major terms used within this guidebook.

1.3.6 Appendix B: Tools and Techniques for Microprocessor Development and Support

Describes tools and techniques unique to microprocessors.

1.3.7 Appendix C: Bibliography

Provides a representative bibliography of professional papers addressing the topics of this guidebook.

2. RELEVANT DOCUMENTS

2.1 REGULATIONS, SPECIFICATIONS AND STANDARDS

The following government documents are relevant to the acquisition of microprocessors and firmware in airborne systems.

- DIRECTIVES

DODD 5000.1	Acquisition of Major Defense Systems
DODD 5000.3	Test and Evaluation
DODD 5000.29	Management of Computer Resources in Major Defense Systems
DODD 5000.31	Interim List of DOD-Approved High-Order Programming Languages

- MILITARY STANDARDS

MIL-STD 483	Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs
MIL-STD 490	Specification Practices
MIL-STD 881A	Work Breakdown Structure
MIL-STD 965	Parts Selection and Control
MIL-STD 1521A	Technical Reviews and Audits for Systems, Equipment, and Computer Programs
MIL-STD 1679 (Navy)	Tactical Software Development
MIL-S-52779A	Software Quality Assurance Program Requirements
MIL-HDBK-217C	Reliability Predictions of Electronic Equipment

- <u>AIR FORCE DOCUMENTS</u>

AFR 26-1	Organic Support
AFR 65-3	Configuration Management
AFR 80-14	Test and Evaluation
AFR 122-9	The Nuclear Safety Cross-Check Analysis and Certification Program for Weapon Systems Software
AFR 122-10	Nuclear Weapon Systems Safety Design and Evaluation Criteria
AFR 300-10	Computer Programming Languages
AFR 800-2	Program Management
AFR 800-6	Program Control-Financial
AFR 800-14	Management of Computer Resources in Systems
AFR 800-24	Acquisition Management of Parts Control Programs
AFSCR Pamphlet 800-3	A Guide for Program Management
AFSCR 800-13	The Management Control of the Selection of Parts in New Equipment Design
AFSC DH4-2	Electronic Systems Test and Evaluation
SAMSO-STD-73-3	Standard Engineering Practices for Computer Software Design and Development

- <u>OTHER GOVERNMENT DOCUMENTS</u>

SECNAVINST 3560.1	Tactical Digital Systems Documentation Standards

2.2 OTHER DOCUMENTS

- <u>Airborne Systems Software Acquisition Engineering Guidebooks</u> - all are applicable to this guidebook.
- SEID Software Product Standards, Feb. 1977, TRW D01445.
- "Embedded Computer Resources and the DSARC Process", Management Steering Committee for Embedded Computer Resources, Barry DeRoze, Chairman, Office of Secretary of Defense, Room 2A318, Pentagon, 1977.

3. GENERAL GUIDELINES FOR MICROPROCESSORS AND FIRMWARE

Microprocessor Large Scale Integration (LSI) technology is not a revolution but an evolution of digital technology to greater gate density, more gates per package, and lower cost per gate. Therefore, the nature of the problem confronting the program manager or engineer has not changed, merely the relative importance of the parameters.

The two key parameters that have changed significantly are component and programmed memory proliferation.

- Component Proliferation - Both the variety and the complexity of LSI components are increasing. The problem is further compounded by the decreased intercompatibility, vendor life cycle support, and testability of the components.

- Programmed Memory Proliferation - Both the variety and quantity of programmed memories in embedded computer subsystems are increasing. This problem is critical to avionics system management because despite the orders-of-magnitude improvement in device costs, there has been *no corresponding progress in the* reduction of life cycle costs for the intelligence contained in the memories, i.e., the software and firmware. The technological trend is for hardware costs to decrease to a less significant percentage of the total embedded computer subsystem cost.

These two problems jointly present the Air Force with a variety of devices which perform critical mission functions but may be impractical to support on either the hardware or software level.

This continuing evolution of digital technology has triggered a revolution in the digital marketplace resulting in larger volume production, a greater variety of configurations, and a greater range of applications. It has also triggered a revolution in the application of digital devices in airborne systems because more airborne functions are being digitized, centralized computers are yielding to distributed systems, and discrete logic (MSI and SSI) is yielding to programmable logic (LSI with firmware).

This technology poses new problems to the Air Force resulting from the unique requirements and constraints of the economics of mass production, the high cost of digital design, and the special requirements of the Air Force.

- Economics of Mass Production
  - Economy depends on mass-production.
  - Mass-production depends on a mass market.
  - Marketability of devices is enhanced by flexibility of application.
  - Digital devices achieve the ultimate in flexibility through functional parameterization on the state of a programmable logic array (memory) - another definition of a microprocessor.

Therefore, economics favor mass-production of microprocessors that are flexible and highly memory intensive, i.e., microprocessors that must be configured and programmed for each application.

- High Cost of Digital Design
  - Both the design of the hardware configuration and the programming of the digital memories are labor intensive, i.e., contain a large number of design decisions.
  - The number of required design decisions for programming memory devices greatly exceeds the number required to interconnect the devices themselves. (For example, a single-chip microprocessor may have 16K bits of internal programmable memory and only 40 pins). Once the devices are interconnected, the programmable memory can provide an almost unlimited degree of flexibility to the system.

Therefore, software labor cost tends to dominate total design cost in most digital subsystems.

- Special Requirements of the Air Force
  - Device Characterization - The rapidly advancing microprocessor technology is still relatively uncharacterized in terms of reliability, hardness, shelf-life, etc.

- Long Life Cycle - The life cycle of a typical airborne system may exceed twenty years; yet, microprocessor economics dictate a rapidly changing selection of components. If the mass market for a selected part subsides, support becomes more expensive. If a new device is substituted, the configuration may have to be redesigned.

- Configuration Management - A baseline must be established embodying unambiguous identification, status accounting, and quality control procedures - yet, a programmable memory component (firmware) has an almost unlimited number of possible internal configurations, all of which can only be detected using special equipment.

- Standardization - From the standpoint of the Air Force, multiple acquisitions serving the same function but with different configurations must be minimized. However, there are a multitude of microprocess families and components, none of which are satisfactory for all conceivable mission requirements, yet many of which overlap extensively. (A contractor may claim his selection is the only one that will suffice). On the other hand, the Air Force has little opportunity to enforce standardization on the mass market, as it is a relatively low volume consumer of devices.

- Logistic Support - Some airborne systems need to be frequently (e.g., mission-to-mission) reprogrammed to meet changing requirements - yet, embedded firmware (which is relatively inaccessible, requires special reprogramming equipment, and is dependent on a supply of firmware devices) is proliferating in such systems.

- Documentation - Satisfying total AFR 800-14 documentation requirements for computer programs is expensive and would tend to hinder the proliferation of programmable microprocessor devices if blindly adhered to - yet, inadequate documentation of critical programs can be disastrous to the system mission. Careful tailoring of documentation requirements is necessary.

SUMMARY OF PROBLEMS

- Proliferation of
  - Micros in airborne systems (anticipated)
  - Micro families in the marketplace
  - Programmable memories
- Expense of
  - Digital design and redesign
  - Full CI/CPCI Documentation
  - Life cycle support
- Difficulty of
  - Configuration control for firmware
  - Device characterization and qualification
  - Logistics and support
  - Staffing qualified personnel
- Dependence on
  - Continuance of mass market for selected devices
  - Vendor support of devices
  - Key personnel in labor-intensive effort
- Conflicts between
  - Volatility of microprocessor market and long life cycle of airborne systems
  - Diversity of microprocessor devices and Air Force desire for standardization
  - Proliferation of firmware devices and Air Force need for documentation, configuration control, and field reprogramming
  - Speed of technological progress and need for mature understanding and policy

3.1 DEFINITION OF TERMS

This section provides a definition of the main terms used in the guidebook.

- Microprocessor - "One or more large scale integration (LSI) devices that, when interconnected, perform the function of a Central Processing Unit (CPU)". (AFR 800-14, Vol. I/AFSC Supplement 1, 8 August 1977, Att. 1, paragraph 14.2). Large scale integration is defined in paragraph 14.8 of the above as "complexity greater than approximately 1000 logic gates". This is preferable to MIL-STD-HDBK 217B which defines it as 100 gates or greater. A current draft of Electronic Industries Association (EIA) definitions describes a microprocessor as "a single integrated circuit which determines and implements at least the arithmetic logic unit, control function, and instruction-set architecture of a computer".

- Microcomputer - "A microprocessor plus other components, such as memories, clocks, and various interface devices that collectively operate as a stored program computer". (op. cit., paragraph 14.3). A simpler definition is a computer whose CPU is a microprocessor. Microcomputers may come packaged on a single chip or set of chips, and often are sold as a preconfigured card or set of cards.

- Firmware - The term firmware is used for two different concepts which must be distinguished. The first is, "Computer programs and computer data at the microprogram level". (op. cit., paragraph 14.5). This type of firmware is concerned primarily with computer design and instruction set definition and implementation. It is not within the purview of this guidebook and will not be considered further. The second is, "Any level of executable computer programs and computer data that cannot be readily modified under program control, that is, read-only" (Ibid.); "Software that resides in a non-volatile medium which is read-only in nature. Firmware is completely write-protected when functioning in its operational mode." (AFR 122-10, 27 November 1978, Att. 1, paragraph 20). Common microprocessor (LSI) firmware devices are shown in Table 3-1.

The above definition stresses the concept of firmware as a computer program in lieu of a hardware device. Other published definitions

Table 3-1. Common Microprocessor Firmware Devices

Type	Abbreviation	Characteristics						Reprogramming Equipment Required
		Programmed by		Reprogrammable		Reprogramming Turn-Around Time		
		Factory	User	Yes	No	Long	Short	
Mask-Programmed Read-Only Memory	(Mask-Programmed) ROM †	X			X	X		User defined mask Factory equipment
Programmable Read-Only Memory	PROM		X		X		X	PROM Programmer
Eraseable Programmable Read-Only Memory	EPROM ‡		X	X			X	Ultra-violet eraser PROM Programmer
Electrically-Alterable Read-Only Memory	EAROM		X	X			X	Special circuitry
Electrically-Eraseable Read-Only Memory	EEROM		X	X			X	Special circuitry

† ROM's typically have the fastest access times of firmware devices.

‡ Note that proposed paragraph 3.1.36.17 to MIL-E-5400 prohibits EPROM's.

which consider the "device" as the firmware and whose use is discouraged are

- "Logic circuits in read-only memory that may be altered by the software under certain circumstances". (AFR 122-9, 1 July 1974, paragraph 3d).
- "The logical code of computer equipment which interprets the control functions of the equipment". (DODD 5000.29, 26 April 1976).
- "Non-volatile semi-conductor memory which contains segments of the system intelligence". (AFR 800-14, Volume II/AFLC Supplement 1, draft).

3.1.1 Hardware-Intensive, Firmware-Intensive, and Software-Intensive Applications

Programmable devices embedded in computer systems range from simple Read-Only Memory (ROM) containing tabular data to microcomputers performing specialized functions to general purpose memories containing central computer flight programs. Current Air Force policy, as contained in AFR 800-14, Volume I/AFSC Supplement 1, paragraph 3m(8), requires all these "computer programs" to be developed and managed identically. While this could be done, it is preferable to divide computer programs into different categories of application for which life cycle management policy can be optimized with regard to cost effectiveness. Current activity in AFSC is underway to bring this about.

The categories of application suggested by this guidebook are termed hardware-intensive, firmware-intensive, and software-intensive, and are defined in terms of the system specification as follows

- A hardware-intensive application is supplied by the contractor to satisfy a hardware specification, is not reprogrammed by the government, and is only reprogrammed or replaced by the contractor in response to a change in the specification.
- A firmware-intensive application is usually a contractor response to a system functional requirement on a prescribed hardware host. The contents of the firmware-intensive application memory is maintained by the contractor, with the government maintaining the equipment configuration. The application memory is modified by the contractor only in response to a change in the functional requirements of the system in which it is embedded.

- A software-intensive application is a response to a specification in which reprogrammability is a system requirement. The entire system (hardware plus firmware) is maintained by the government. Full CPCI documentation is provided by the contractor.

The distinguishing characteristics of the three categories are shown in Table 3-2.

3.1.2 Examples of Hardware/Firmware/Software-Intensive Applications

- Hardware-Intensive

  - A firmware device used to hold microinstructions for a flight computer. The device is used by the contractor merely for design convenience in the implementation of the instruction-set architecture of the computer. The system functional requirements are implemented via the instruction-set architecture of the computer, not the microinstructions.

  - A microprocessor embedded in an anti-skid device which can be fully tested and will not have its computer program modified unless the anti-skid device is completely redesigned. No enhancement of the program is intended after completion of the development and test of the device. Only reprocurement package may be bought.

  - An inertial navigation system with an embedded computer interfacing through an aircraft standard MUX bus for which no DOD maintenance is necessary. The contractor will offer a warranty for system performance. If errors or performance improvements are necessary the contractor provides the new units or modifies existing units under the warranty.

  - A microprocessor embedded within a terminal to generate characters on a display which will have the capacity desired for all planned display activity for the future. No support for the microprocessor computer program is planned. If an unplanned change is required, the work will be contracted out on a one time basis.

  - A microprocessor embedded as a fusing device of an artillery shell. The details of the weapon hardware strongly influence the architecture of the digital fuse and production economics. Once the computer program and shell are tested a number of shells are issued to the field. A change to the computer program on those released is much too expensive to justify except under extreme circumstances. Product improvements on yet to be made rounds might take place, but after production is complete no change will be made.

Table 3-2. Distinguishing Characteristics of Hardware/Firmware/Software-Intensive Applications

Application	Characteristics	Examples
Hardware-intensive	• Tabular data, truth tables, highly-embedded, fixed function	Character-generator PLA Instruction-set microcode ROM†
	• Implementing the hardware that implements the required function	Embedded control firmware in commerical "widgets" (e.g., disk controllers)
Firmware-intensive	• Contractor-supplied firmware for government equipment	ROM-based support software: • translators • editors • loaders, etc
Software-intensive	• Mission-reprogrammable	Operational flight programs, mission data ROM's
	• Change-intensive, government reprogrammed	ATE firmware

† High level instructions, e.g., sine/cosine, might fall into the firmware-intensive category if contractor supplies them and no basic hardware changes are required.

- Firmware-Intensive

    - Firmware-based support software package for a commercial microcomputer development system. The package is supplied to perform to functional requirements specified by the government. Additional packages are planned for hosting on the same development system. Maintenance of the package is provided via contract.

- Software-Intensive

    - Firmware-driven Automatic Test Equipment (ATE) - the firmware is regularly redesigned at a central government facility and released to fielded ATE units. Note that if the firmware is provided by contractors for government-owned ATE the application becomes firmware-intensive.

    - A mission-reprogrammable airborne radar warning receiver. The flight program is firmware-based and is divided into separable code and data portions. The system hardware and code are supported by the government with a full-time staff, and the data portion is reprogrammable with special fielded reprogramming stations. In normal operation the field users modify the data portion as required to respond to changing enemy threats and aircraft configuration changes. This type of system requires full AFR 800-14 management control and documentation.

3.2 ACQUISITION CONSIDERATIONS

General recommendations for consideration during system acquisition are as follows:

1. Exploit the economics of the microprocessor marketplace. Since the microprocessor is a mass-produced product, considerable cost savings can result from making maximum use of the microprocessor marketplace.

    - Acquire the marketplace "winners" - the most popular devices are the most likely to be available and supported throughout the system life cycle.

    - Try to make what already exists work before paying for a new development. Consider general purpose machines for special purpose applications before paying for new, special purpose machines. (For example, consider double precision arithmetic on n-bit machines before paying for new, unproven 2n-bit hardware). Use commercial cards instead of customizing cards unnecessarily.

2. Consider excess hardware capability (memory, throughput, I/O) when possible to cut software costs. Trade studies should include at least the following reasons for excess capacity

- Provides growth margin - Reduces deficiency corrections to software-only changes; delays obsolescence of hardware.
- Needed to support Higher Order Languages (HOL's)
- Provides more "bells and whistles", redundancy, error checking.
- Discourages tricky code (programming costs increase geometrically as processor capacity is approached).

3. Consider flexible, reprogrammable firmware devices when

- They fulfill all other mission requirements (speed, hardness, reliability, size costs, etc.).
- They are expected to be available in sufficient quantities over the system life cycle.
- They are supportable (necessary reprogramming equipment available, etc.).
- They are needed (application not hardware-intensive).

4. When evaluating use of non-reprogrammable firmware devices (e.g., mask-programmed ROM)

- Insure availability of the mask.
- Confirm that the programming mask can be transmitted to another manufacturer if necessary.
- Establish a mask library if necessary.
- Consider buying a life cycle supply of parts at the outset in light of shelf life, impact of reprogramming need, storage cost, and projected logistics needs.

5. Evaluate the development/support system along with the microprocessor; each impacts the cost-effectiveness of the other. The

Microprocessor Development System is discussed in Appendix B. Categories to consider in a proposed microprocessor development system are

- Language Translators (i.e., compiler, assembler, interpreter)
  - Are they commercially maintained and/or organically maintainable?
  - How mature are they?
  - Do they provide required functions?
  - Where have they been hosted?
  - Are they transportable?
  - Whose else uses them?
  - How well are they documented?
- Mass storage
- Operating System
- Documentation
- Ease of use
- Extendability
- What activity level does it support? (e.g., number of users)
- Vertical compatibilities with company product line
  - How does it support multi-processor systems?
  - PROM Programmer
  - ICE
  - MODEM
  - Logic Analyzer
  - Other computers
- Reliability, availability, maintainability

6. Investigate the manufacturer along with the product.

- Will he "be there" for the duration of the life cycle of your system? The cost to your system of loss of mass production support for your devices may be great.
- Field support - Especially important if the product is card-level or higher.
- Warranties
- Service options
- Second sources

7. For software-intensive applications, get delivery of the data and the equipment necessary to reprogram and revalidate.

- Secure right to use - Required by AFR 800-14, Vol.I/AFSC Supplement 1, paragraph 3i.
- Consider contractual provisions for long term support.
- Establish backups, safeguards, e.g., second sources, IV&V.

3.3 MANAGEMENT CONSIDERATIONS

3.3.1 Firmware Concepts

The key concept in firmware management is to identify, classify, and formulate a life cycle support concept as early as possible. Life cycle trade studies may be required to determine the support concept. This classification should be coordinated with the Computer Resource Working Group (CRWG) and periodically reviewed throughout the life cycle and changed if necessary. The CRWG should play a major role in determining the life cycle support concept.

- Identification
  - A statement in the RFP requiring the bidder to identify each programmable memory in the proposal may help discourage the "hiding" of firmware subsystems by contractors who wish to avoid expensive management controls.

- Classification
  - Each memory will have to be examined in relation to the following factors
    - Number and type of users - mass produced commercial firmware is usually classifiable as hardware or firmware intensive.
    - Anticipated need for reprogramming - operationally reprogrammable memories are classifiable as software intensive.
    - Firmware medium - media with long reprogramming turnaround times (such as mask ROM) are not ordinarily classified as software intensive.
    - Complexity of code - this primarily distinguishes hardware intensive from firmware intensive.
- Support Concept
  - Hardware-intensive memories (as programmed) can be handled as piece parts in equipment CI's. As such, supporting a hardware-intensive memory is little different than supporting any microprocessor component.
  - Firmware-intensive memories are normally supported by the contractor. Establish both the ability and the commitment of the contractor to support them.
  - Software-intensive memories are ordinarily supported organically. Full CPCI documentation, rights to use of support equipment (including reprogramming development system and software), and in general, all the weight of AFR 800-14 falls on the development.

3.3.2 Personnel Considerations

3.3.2.1 Training

The Computer Resources Product Division focal points establish training and education programs for managers and engineers in the management technical areas of computer resources, and the Directors of Procurement and Manufacturing insure that procurement personnel directly involved with the acquisition of computer resources maintain a high level of expertise on acquisition of computer resources (AFR 800-14, Vol. I/AFSC Supplement 1, paragraph 5b (1a) and (4)).

The microprocessor industry is so fast-moving and dynamic that this requirement can only be satisfied by continuing education programs. The main areas of training are shown in Table 3-3.

3.3.2.2 Staffing

The integrated nature of microprocessor configurations defies the traditional hardware/software split in specialization. Specialization to some extent is still necessary, but interface between true specialists is becoming increasingly difficult. Therefore, the program office should place emphasis on staffing those with general digital skills as well as strictly hardware or software skills.

3.3.3 Microprocessor Programming Languages

Current Air Force policy (AFR 800-14, Vol. I/AFSC Supplement 1, paragraph 3e(1)) is that Air Force approved Higher Order programming languages will be used to develop computer programs unless none of them are cost-effective or technically practical over the system life cycle. Requests for waivers (as required by AFR 300-10, paragraph 9) will be submitted through the product division computer resources focal point. (In the case of hardware intensive applications, AFSC is formulating policy which should eliminate the need for waivers). If waivers are necessary, Table 3-4 provides guidance for consideration.

Some key tradeoffs between popular microprocessor HOL's are

- FORTRAN - Basically a scientifically oriented floating-point language, it requires a large memory overhead in system routines to perform mathematical functions and I/O. As such, it is very inefficient on machines with small word width, small memories, and fixed-point arithmetic units. For processors that can support it, it is one of the easiest languages to optimize for speed. Supports modularity and can be made to support structured programming (e.g., IFTRAN, etc.). (FORTRAN is included in the AFR 300-10 list of approved HOL's).

- JOVIAL J73 - A block-structured language similar to PL/1 and PASCAL. Compilers are currently being developed for several microprocessors. (Also an AF approved language).

Table 3-3. Training for Microprocessors and Firmware

<table>
<tr><th>Hardware Area</th><th>Software Area</th></tr>
<tr>
<td>

LSI Technology

Fabrication
Circuit theory
Logic families
Memory technology
Military qualification
Characteristics of industry and marketplace

Computer Architecture and Theory of Operation

Fixed/bit-slice architectures
Instruction sets
Bus structures
Timing and control
LSI support chips

Memory
I/O
Programmed Logic Arrays
D/A and A/D converters

Hardware/software/firmware partitioning tradeoffs
Mass storage devices
Peripherals
Military/Industrial standards

</td>
<td>

Data structures and algorithms
Machine/assembly language
HOL's
Microprogramming

Vertical
Horizontal
Emulation

Support software

Operating systems
Language translators

Compilers
Assemblers
Interpreters
Linking loaders
Object formatters

Cross-software

Cross-assemblers
Cross-compilers

Programming practices and standards
Documentation

</td>
</tr>
<tr><th colspan="2">Support Area</th></tr>
<tr>
<td colspan="2">

Turn on, turn off, and system operation
Periodic maintenance requirements and procedures
Operating and maintenance documentation familiarization
Interface with other two areas

</td>
</tr>
</table>

Table 3-4. Waivers for Programming Languages

Attempted Justification	Remark
Compiler Unavailability	This is not necessarily a good reason for granting a waiver. Possible responses include use of another microprocessor that has one, or development of a compiler. (Budget and schedule impacts should be thoroughly investigated and included in the request for waiver).
Performance degradation by HOL code	The use of more computing resources (better microprocessor, more memory) should be seriously considered. Given the typical high ratio of software to hardware costs total costs may benefit.
Excessive resources (time and equipment) needed to compile	The availability of large timeshare systems, minicomputers, and microcomputer development systems, LCC effectiveness of HOL's, rapid advance of computer technology and the need for life cycle maintainability make this attempted justification weak.
Insufficient code to justify HOL costs (compiler, training, resources, etc.)	This will have to be evaluated on its own merits for each particular case. The support concept also must be considered here.
Increased difficulty of debug (isolation of programmer from object code)	An objection of little weight, especially if an MDS with an ICE and symbolic debug capability can be acquired (see Appendix B).

- BASIC† - This is (usually) an interpretive language, meaning that each statement is compiled as it is keyed in, compilation usually consisting of compression of the statement. Numbers are held as strings of binary-coded decimal (BCD) characters. This is one of the most flexible languages available, but one of the slowest to execute. This language does not support structured programming, modularity, or fixed-point arithmetic.

- PL/M† - This PL/1-like language, designed to support microprocessors, has data types natural to them. Comparatively little support is available with this language because of its immaturity. Supports block structure modularity, structured programming, and fixed-point arithmetic.

- PASCAL† - Another block-structured language with wide university support. Includes the capability to define new data types. Supports modularity, structured programming, etc.

- ADA - The proposed DOD standard. This language was designed with embedded applications in mind, however, the full language may be too much for some microcomputers to handle. A reasonable subset will have to be defined in the light of unique microprocessor characteristics and constraints.

When a Low Order Language (LOL) such as assembly language must be used, focus on the software technique by stressing the formulation and implementation of a disciplined, structured programming and development methodology. Some applicable techniques include

- A structured design language
- Coding standards and conventions
- Modularity
- Library control
- The use of macro-instruction sets

Well-structured LOL might be amenable to translation to HOL during a later phase in the life cycle when the HOL becomes feasible.

---

† Use of these languages for deliverable code requires the appropriate waiver.

Remember that there are several goals in using higher order programming languages

- Productivity
- Less source lines of code to implement, therefore easier to understand and support
- Transportability of code (machine independence)
- Reusability of code (avoids duplication of effort)

3.4 MICROPROCESSORS AND COMPONENTS

3.4.1 Management and Control

AFR 800-14, Vol. I/AFSC Supplement 1, paragraph 3e(2), states that microprocessor devices

- Are to be treated as microelectronic circuits for parts selection and control.
- Are to be selected using an AFSCR 800-13/AFR 800-24 Program Parts Selection List and Parts Control Board.
- Are to be controlled by a MIL-STD-965 Parts Control Program.
- Which have not been qualified to the MIL-M-38510 series of microcircuit design specifications should be coordinated with the product division focal point.

For each microcircuit used by a contractor which is not qualified to the requirements of MIL-M-38510

- A specification should be prepared by the contractor which completely satisfies the form, fit, and performance (including reliability and longevity) of the microcircuit requirement.
- The specification should be in a format similar to a MIL-M-38510 slash sheet, except that the specification number and FSC designation should be omitted (a contractor numbering system should be used).
- The specification should satisfy applicable requirements of MIL-M-38510.

3.4.2 Parts Selection

Parts selection must take into account life cycle parts control. No part should be selected without determining its life cycle support concept. Some considerations are

- Availability - Is the part to be procured as needed during the system life cycle, or is a life cycle supply of parts to be procured at the outset?
    - Procured as Needed - Is the part popular? Are second sources available during the life cycle? Are there any contractual commitments to life cycle availability on the part of the manufacturer? Can the manufacturing process be transmitted to another manufacturer if necessary?
    - Procured at Outset - What is the shelf life of the parts? How are they to be stored, and what will it cost?
- Reliability - Are the failure rates for each part known? Are the reliability prediction methods relevant to the mission environment?
- Maintainability - Are parts accessible in the system? Are higher failure rate parts more accessible than lower failure rate parts? Are reprogrammable memory components easily replaceable, e.g., mounted in sockets when feasible? Is the microprocessor connection designed to allow ICE plug-in?
- Logistics - Can parts be distributed to support centers when needed? Can software-intensive memories be reprogrammed and distributed as needed? Are firmware parts segregated and organized on the minimum number of cards? What field-shop equipment is required?
- Identification - Can all configuration items be identified and documented accordingly? Can firmware components be identified internally and externally?

When several qualified parts exist, the economics of mass-production are very favorable to going with the marketplace "winners". Not only hardware, but software costs will benefit, because the most popular parts will generally have the broadest software base, best documentation, largest number of qualified programmers, and probably the largest selection of support tools.

3.4.3 Standardization

Standardization is applicable to a number of aspects of system design

- Interfaces
    - Electrical/Physical
    - Bus protocol
    - Module partitioning
- Microprocessor family, e.g., 8-bit, bit-slice
- Microprocessor instruction-set architecture
- Development/support system and tools
- Programming Languages

Advantages and disadvantages of parts standardization are shown in Table 3-5.

Table 3-5. Parts Standardization - Advantages, Disadvantages

Advantages	Disadvantages
• Improved logistics and supply	• Reduced performance in some applications
• Reduced parts costs	• Inappropriate architecture encourages "tricky" software
• Single programming language	• Forces technological progress into vertical compatibility straight-jacket
• Concentration of resources (software, training, documentation)	

Standardization, to be worthwhile, must reduce the total life cycle cost for the systems involved, yet the existence of a competitive mass-market is a key factor in cost reduction. It has been suggested that standardization of interfaces is more desirable than standardization of technology. (See Bibliography, "AVIONICS: The Road Ahead").

3.4.4 Reliability

MIL-HDBK-217C is the source of reliability prediction data and models for microelectronic devices. From these data, it is clear that IC package count, pin count, and component misapplication (more prevalent for MSI and SSI parts) contribute more to failure rate than chip complexity (gate count). Hence, an LSI circuit may have a lower hardware failure rate than an equivalent MSI or SSI implementation.

Properly used, LSI technology can also increase software reliability. This is because distribution of function between several memories may enhance the simplicity of each enough to counter the impact of a larger total amount of software.

In any case, bear in mind that software reliability can be increased throughout the system life cycle, while hardware items are subject to eventual wearout.

4. SPECIFIC GUIDANCE FOR MICROPROCESSORS AND FIRMWARE

4.1 MANAGEMENT, PLANNING, AND CONTROL

4.1.1 Program Management Responsibilities

AFR 800-14, Volume 1/AFSC Supplement 1 establishes policy for the acquisition and support of computer equipment and computer programs employed as dedicated elements, subsystems, or components of systems developed or acquired under the management concept established in AFR 800-2. Paragraph (5) specifies that microprocessors, microcomputers, and firmware will be considered computer resources. Other pertinent considerations are

- Computer (including microprocessor) resource requirements will be reviewed during the Conceptual and Validation phases of system acquisition. (paragraph 3a).
- During these phases, risk analysis, life cycle planning, preliminary design, security features, and interface control methodology will be addressed, particularly emphasizing the integration of computer programs into the system. (paragraph 3a).
- The Computer Program Development Plan (CPDP) should be used to define and monitor specific milestones to manage the life cycle development of computer resources, including support computer programs, and used to ensure that the proper development activities are accomplished.
- These activities include planning, analysis, design, implementation, integration, test, documentation, operation, maintenance, and modification. (paragraph 3m(10)).
- A Computer Resources Integrated Support Plan (CRISP) will be developed for each system acquisition, before Milestone II Review or equivalent. (paragraph 3m(5)).

- The preparation of the CRISP and administration of the CRWG will be provided for in the Program Management Plan (PMP). The Program Management Directive (PMD) should assign the responsibility for CRISP preparation to one organization. (paragraph 3m(5)).

- A Computer Resource Working Group (CRWG) will be formed for each system acquisition, and its responsibilities defined before the Full-Scale Development Phase. (paragraph 3m(5)).

4.1.2 Planning

Initial planning should strive to identify all programmable memories in the system, including its support system, and the life cycle support concept for each.

- Beware of these common pitfalls
    - Insufficient lead times for parts procurements
    - Inadequate development/support tools
    - Inappropriate personnel requirements
    - Lack of control of subcontractors
    - Lack of attention to standardization considerations

- The Program Office should include "general digital" people (see Section 3.3.2.2) in order to evaluate hardware/software/firmware tradeoffs early for adequacy, completeness, simplicity, and testability before too many more resources are committed.

- Program milestones should be clearly conceived, especially in relation to how they are to be monitored for attainment. Identification and documentation requirements should be determined in relation to a configuration management concept.

- IV&V concurrent with the development effort should be considered, and if decided on, appropriate contractual provisions made to assure timely delivery of information (including draft documentation) to the IV&V contractor.

4.1.3 Establishing Computer Resource Requirements

The definition and specification of requirements for microprocessors and firmware involve all the considerations for computers given in AFR 800-14, Volume II, Chapter 3. These, with unique considerations for microprocessors and firmware, are presented here.

Considerations in initiating computer resource requirements

- Required capabilities
- General purpose versus special purpose computers
- Operational availability
- Compatibility with existing systems
- Preferred programming language and accompanying rationale
- Flexibility and growth
- Training for support
- Organic support
- Support concept
- Existing computer resources
- Security
- Hardware/software/tradeoffs
- Central/distributed computer tradeoffs
- Verification and Validation
- Environment

Considerations in establishing minimum computer design characteristics

- Cycle time (processor and memory)
- Word length (data and address)
- Memory size, type, characteristics

- Arithmetic capability - fixed point, floating point (software or hardware)
- Multiprocessor, multicomputer, single computer configurations
- I/O channels and configuration, transfer rates and interrupts
- Power fail safe
- Parity checks
- Internal timers and real time/line time clocks
- Address and instruction traps
- Number and types of registers
- Instruction repertoire
- Peripheral equipment
- Off-line backup storage capacity
- On-line mass storage devices
- Communication options
- Interrupt structure
- Sequential logic/microprogram control
- Equipment redundancy
- Facility impact
- Electromagnetic interference
- Security

Architectural trends in microprocessor systems

- Dynamic Configurations
    - Multi-processor
    - Shared facilities
        - Memories
        - I/O

- Distributed logic
- Embedded processors

- More Memory-Driven Logic
  - Separate data and program memories
  - Internal control (micro-program) memories
  - Buffer memories
  - Larger memories
  - More memory types

- Closer Coupling Between Digital Processors and Their Environment
  - More interface channels
  - Higher data rates (communication bandwidth)
  - More complex protocols
  - Distributed communication logic
  - More complex interdependence (timing, control)

Factors to consider when selecting a microprocessor (in order of importance)

- Physical characteristics - must satisfy mission requirements (this is the bottom-line requirement).
- Overall throughput for intended application (cycle time of memory and processor, architecture, instruction set, etc., interact with the application so intimately that overall throughput when executing the intended application is the only valid measurement).
- Support software and hardware - this will probably be the biggest discriminator among competitive microprocessors, hence a high-leverage item.
- User base - "marketplace winner" factor.

- Family compatibility and upgradeability - a leverage item for high velocity technology.
- Maintenance and field service - important for card and higher level products.
- Supply and support - an important logistics consideration, e.g., is device second-sourced?
- Previous experience (including reputation) - an additional discriminant.
- Cost - listed last because of the minor nature of device cost compared to system cost.

Microprocessor system design goals for life cycle cost minimization

- Physically segregate software-intensive and firmware-intensive firmware onto the minimum number of cards for easy replacement and to simplify logistics and hardware change.
- Design the microprocessor to be accessible to ICE plug-in. May require special design for soldered-in micros.
- Design-in probe points for logic analysis and signature analysis.
- Design reprogrammable data bases to minimize the parts-cost of reprogramming.
    - Organize memory chips so that full memory words reside on a single chip - this minimizes the number of chips affected by a reprogramming change.
    - Separate the reprogrammable data base from the operational code (also enhances security by permitting separate security classifications for code and data).
    - For PROM's that can be overwritten, begin valid data tables with a flag word equal to the unprogrammed state of the part (all 1's or all 0's): to reprogram, overwrite the flag word with a pointer (not all 1's or all 0's) to a new table in an unused area of the same part.
    - Address frequently reprogrammed tables via indirect addresses held in a localized, inexpensively-reprogrammed area of memory.

- Provide test programs in and for all firmware where feasible
  - Self-resident functional-level tests (GO/NO-GO, etc.) for hardware-intensive memories
  - Diagnostics for firmware-intensive memories
  - Maintenance programs resident in AGE or ICE for software-intensive memories
  - Emphasize fault isolation to board level
- Leave spare capacity wherever possible
  - Modular interfaces
  - Connectors
  - Cards
  - Sockets
  - Memories
    - Spare addressing capacity
    - Spare memory per chip
    - Spare chips
    - Spare access time
  - Throughput
  - I/O (channels and band width)
- Design interfaces to facilitate component upgrade
  - Design memory interface to permit mixed memory types
  - Standardize I/O protocol to permit device substitution

Considerations in establishing minimum computer program design requirements

- Executive/supervisor features
- Modularity

- Program types
- Minimum iteration rates
- Program media
- Accuracy and stability
- Fixed point versus floating point arithmetic
- Security
- Existing programs available
- Restart, restore, and recovery
- Support software features
- Testability

To these may be added

Support software considerations

- Support software features including language translators (assemblers, interpreters, compilers) and operating systems
- Support software host (same as target machine or different)
- Suitability of support software host for life cycle maintenance of target system
- Capability of hardware support (ICE and PROM programmer) with support software host
- Suitability of support software host for program and documentation configuration management control
- Obsolescence
- Evolutionarity (upgradeability of hardware and software, portability of software)
- Maintainability

4.1.4 Documentation Requirements

In general, any digital system acquired by the Air Force should be fully documented. The Hardware-Intensive/Firmware-Intensive/Software-Intensive concept is of use only in defining what formal documentation is deliverable - even though non-deliverable, the contractor should maintain the equivalent documentation in internal form and format, but never in "corporate memory".

Good documentation for a programmable embedded computer system will cover the following system aspects (reference MIL-STD 1521A (USAF), June 1976)

- Hardware design
  1. Circuit diagrams
  2. Clock and timing data
  3. Memory design
     a. Type
     b. Physical characteristics (including access times)
     c. Logical characteristics (including word length, number of words, error detection and correction)
  4. I/O design
     a. Physical characteristics (bus design, timing, current and voltage levels)
     b. Logical characteristics (path widths, program addressability)
  5. Hardware/software interface
- Software design
  6. a. Algorithms
     b. Logic flow

c. Source listing

d. Object listing

e. PROM programmer input listing

7. Design tools

a. Support software

b. Development systems

c. Test and diagnostic tools

Microprocessors typically have complex memory structures for which complete documentation is essential. Some things to investigate are

- Physical address space - what types of chips implement which addresses, what are their access characteristics? For example, read only or read write.

- Logical address space - what addresses are available, what is the behavior of each memory location, is there any memory-mapped I/O, are there any "holes" in the address space?

- Anomalous behavior - what is access behavior at unused addresses, what side effects, undefined conditions, etc., are there? Can two or more processors access the same memory? What about interlock protection?

- Unprogrammed state of the firmware device - all 1's or all 0's?

- Patches - can firmware be overwritten or just programmed once? Can it be erased and reprogrammed? Are firmware components soldered in place or in sockets for easy removal?

- Firmware identification procedures - external (ink stamp, tape, tag), internal (electrically readable).

- Step-by-step procedure, including all operator commands and actions, to generate firmware, erase, and reprogram.

4.1.5 Firmware Data Item Descriptions

For software-intensive application, currently existing software DID's defined in Table 3-1 of the Guidebook for Computer Program Documentation Requirements are adequate. For firmware-intensive application, deliverable documentation should be a reasonable, user-oriented subset of software-intensive documentation. User's Manuals, flowcharts, interface documents, etc., are usually sufficient. For hardware-intensive applications, what is needed is a DID treating a programmed firmware device as a hardware CI. No such DID exists, however, some draft DID's are discussed in the Bibliography. (see "White Paper on AFSC Microprocessor Policy").

4.2 SUPPORT

4.2.1 Summary of Existing Policy

A summary of Air Force support policy, taken from AFR 800-14, Volume II, Chapter 10, is as follows:

Objectives
• Minimum cost • Consultation with user • Expeditious deficiency correction • Cost and mission effective state-of-the-art improvements
Planning
• Implementing command develops CRISP through a CRWG • Using command develops Maintenance Concept in accordance with AFR 66-14. • Supporting command develops ILSP in accordance with AFR 800-8. • All commands identify resources and documentation necessary to support the system. • Periodic assessments will be made to determine the optimum support method (organic, contractor, or both) as per AFR 26-12. The CRISP will be continually updated.
Interservice Support
• Utilize existing DOD resources to maximum extent. • Support decisions must consider life cycle costs. • Interservice support agreements to be approved at Major Command level. • Cross servicing agreements financing will be determined by attaining agency. • Host tenant agreements to be governed by AFR 172-3/AR 37-19/ SECNAVIST 7020.4B.

4.2.2 Impact of Microprocessors and Firmware

Periodic support assessments may need to address the following issues

- Have the status of any programmed memories changed? Things to monitor
    - Hardware and Firmware Intensive Applications
        - Parts vendors (Is part still for sale? Are these second-sources? Replacements/compatibles?)
        - Contractor (On contract to support it? Ability to support it?)
        - Market (Popularity? Obsolescence?)
    - Software Intensive Applications
        - All of the above
        - Application (Need for flexibility decreased?)
        - Organic (Human resources being maintained?)
- Are new cost-effective tools or parts available that increase the support capability consistent with the mission requirements?
- Can standardizations or consolidations be made?

4.3 CONFIGURATION AND CHANGE CONTROL

4.3.1 Environment

AFR 800-14, Volume II, Section 6-4(b) states that computer programs will be managed as essential system elements using common procedures tailored to recognize their unique properties.

- Unique Properties - The unique property of firmware is that the program, the medium, and the coupling of the two must be subjected to configuration management. Management must also take into account that extra facilities beyond the embedded computer may be required to develop the firmware. The unique aspect of reprogramming firmware components is that a single part can have an almost infinite number of configurations depending only on the fixed electrical code image inside it. The same part can be made into a "new" one simply by reprogramming.

- Common Procedures - Common procedures can be established in the following categories
  - Configuration/change control for the firmware.
  - Configuration/change control for the physical firmware devices and end items.
  - Configuration/change control for the firmware development facilities.
  - Centralized reprogramming facility connected to remote PROM burning and testing facilities.
- Essential System Elements - All procedures established for identification, documentation, status accounting, and change control for firmware must recognize them as essential system elements. AFR 800-14, Volume II, Chapter 6 is of use here. The absence of the need for full AFR 800-14 documentation for hardware and firmware intensive applications is the only exception and that exception is currently pending policy modification.

4.3.2 Identification†

The program office should insure that AFLC issues CPIN's for all CPCI's in a configuration-controlled system (AFR 800-14, Volume II, Section 6-5) or secure a waiver. For firmware, however, identification is not complete until the hardware embodying the software is also identified. The hardware and the software inside it must be identified as a unit in order to establish a meaningful baseline. An example of reasonable firmware identification requirements based on DI-A-30001 is shown in Table 4-1.

† See also Guidebook on Requirements Analysis and Specification, Section 5.4.

Table 4-1. Firmware Identification Requirements

Media and Related Documentation
• CPIN • Version/distribution date • Chip number-of-total (e.g., chip 4 of 5) • Socket number • Additional as required (e.g., inventory number, internal check sum)
External Identification
• Vendor part number • Version number • Socket number
Internal Identification (encoded in source text format, near the beginning of the chip if possible)
• Same as media and external identification • Additional as required

An external/internal identification scheme is illustrated in Figure 4-1. Here the internal identification is located near the beginning of each chip, and jumped-around if in the program stream. For chips where this is not possible, the ID location may be changed, or the internal ID requirement waived (checksums are still possible). Gummed labels on program media are governed by MIL-P-83497. Color or bar coding of firmware components needs to be investigated as a means of external identification.

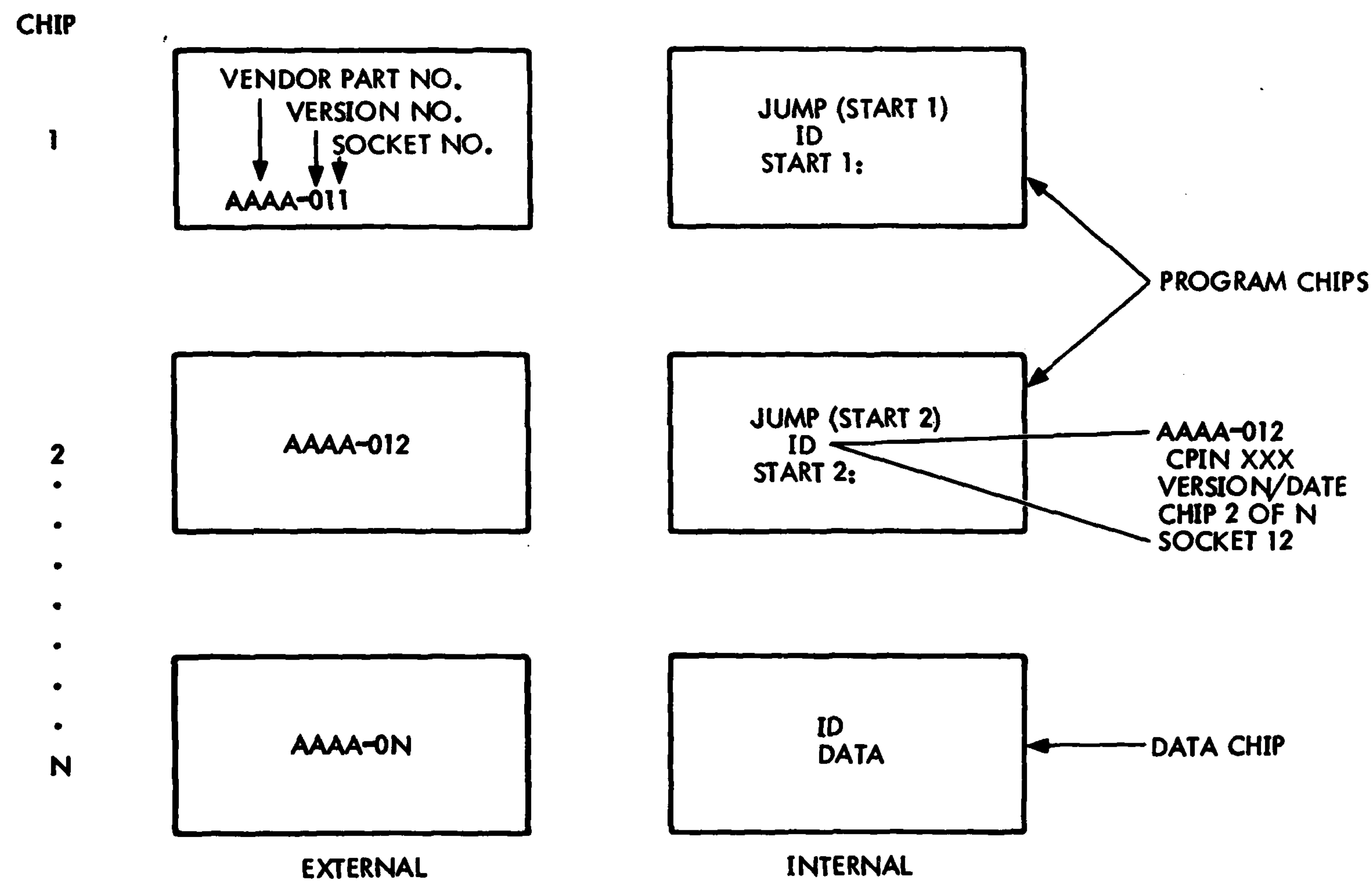


Figure 4-1. Firmware Device Identification

4.3.3 Security†

Firmware memories containing classified information are classified the same as the information. This also applies to the development systems including the PROM Programmers if the memories in them can contain any such information. Methods to erase or obliterate secure codes from firmware devices (as per AFR 122-10, 42d) should be investigated.

4.3.4 Status Accounting

Firmware and firmware devices may be changed on any of a number of levels, including replacement of the physical device (e.g., for a more reliable version containing identical firmware) or by changing the firmware while retaining the original device. A change could also affect both the device and the firmware within the device. Status accounting procedures must be established for each of these levels.

- At the part level
  - With identical parts
  - With compatible replacement parts (e.g., 8080A for 8080)
  - With identical/compatible parts containing a different internal configuration (firmware contents)
- At the level of the firmware contents
  - Optional memory modules
  - Program patches
- At the level of the software program defining the firmware image
  - Specifications
  - The source code file
  - The relocatable object code module files
  - The absolute linked object file
  - The PROM Programmer file

† See also Guidebook on Requirements Analysis and Specification, Section 5.9.

Status accounting procedures must be established for each of these levels if change control is to be instituted. These depend on identification procedures as discussed in Section 4.3.2.

4.3.5 Quality Assurance

In addition to the QA requirements of MIL-S 52779A, the variety of forms a firmware design must go through from source code to parts inventory, necessitate QA checks at each transition point. Automation of these checks is desirable at all levels of software and hardware. At the hardware level, firmware memory contents (total and/or check sum) can be verified automatically, internal ID checks made, and even automatic comparison of external with internal ID accomplished. Configuration control is also desirable for each form or format of the firmware, from HOL source code down.

4.4 TESTING

The principles of computer program testing and evaluation are covered in the Guidebook for Software Testing and Evaluation. Some additional considerations relative to microprocessors and firmware are covered in the following section.

4.4.1 Requirements

It is vital to begin specifying test requirements for microprocessors as early in the life cycle as possible, preferably in the RFP. Typical requirement areas are

- Levels and stages of testability (system build-up procedures)
- Equipment, facility, and personnel requirements
- Test points
  - Where
  - What kind of interface (analog, digital)
  - Speed
  - Automatic or manual

    - Data collection requirements
    - Display requirements
    - Human interface
    - Probe requirements
- Retestability
    - Turnaround time
    - Resource requirements
    - Field tests
- Test case data
    - What kinds and amount?
    - How to generate?
    - What form of storage?
- Automatic test equipment
- Accessibility of parts to test equipment
- ROM simulation capability
- Turnaround time for program source and object patch
- Symbolic debug facilities
- Pre-programmed tests
- PROM Programmer requirements
- Accessibility/modifiability of firmware
- Cost of modifying firmware at each stage of the life cycle
- Adaptability of system executive routine to alternate test modes
- Availability of program stubs for top-down testing
- Self-testability

The Software oriented requirements as listed in the Guidebook on Quality Assurance, Appendix A, are also applicable.

A basic difference exists in test requirements for hardware-intensive memories as opposed to software-intensive memories. For hardware-intensive applications the memory is just a device chosen by the contractor for its operational characteristics; the software is just a design technique. Therefore, if a programming error is discovered, either the system must be reprogrammed by the vendor or a work-around procedure established. Software-intensive memories, on the other hand, are designed to be reprogrammable. Test requirements for hardware-intensive memories, therefore, need only concentrate on the functions performed by the device, sufficient to trace the problem to the device, while for software-intensive memories test procedures must trace the problem to points in the source code.

4.4.2 Self-Testing†

All programmed memories in an embedded computer subsystem should be acquired with adequate self-test programs. "Self-tests" are programs that exercise a set of controllable functions in a systematic and orderly fashion. They are of benefit whether or not modern microprocessor tools such as described in Appendix B are available; in fact, they can be used to indicate when a relatively inaccessible subsystem must be more closely studied.

With modern microprocessor tools that can perform memory-mapping, "self-test" programs can be resident inside or outside the target system

Inside	Outside
Permanently resident in firmware (built-in tests)	Resident in ROM simulator and memory mapped into the system hardware
Temporarily resident (plugged-in for the test only)	Resident in external ROM's and switched in place of the internal memories
Loaded into RAM from AGE	

† See also Guidebook on Requirement Analysis and Specification, Section 4.8.

A variety of different self-tests should be developed for use during all phases of the system life cycle

- Initial Design - To debug the prototype. To establish design/requirements compatibility.
- Development - To isolate system faults.
- Testing - To insure that hardware is operational. To consolidate gains made during previous tests.
- Qualification - To exercise the hardware under controlled environmental conditions to detect intermittent failures that might otherwise become evident only with usage.
- Deployment - To establish a minimal set of working functions. To assist in maintenance and reprogramming. To assist in configuration control (self-computation of internal check sum, etc.).
- Mission - To detect system failures so that informative, corrective, or protective action can be taken.
- V&V - To establish that design specifications have been correctly implemented to satisfy the mission requirements.

ROM's may also be used to stimulate a processing chain with known inputs to establish a baseline level of performance. This kind of test would be desirable for systems which cannot be accessed easily (as on a runway).

The continuing trend toward hardware cost decline might make feasible a "throw away" repair philosophy for faulty circuit cards. In this event, self-tests would need only isolate hardware failures to the card level.

APPENDIX A

GLOSSARY

1. Bit-Slice. A microprocessor whose word-width architecture may be extended by parallel connection with others of the same type.

2. Bus. The communication lines between digital devices.

3. Byte. A group of bits, i.e., 4, 6, or 8 bits depending on the system.

4. Chip. An integrated circuit.

5. Computer Monitor and Control. A (usually interactive) device for monitoring and controlling a computer.

6. Central Processing Unit. That part of a computer which interprets an instruction stream to perform arithmetic and logical functions on a data stream.

7. Disassembly. Translation of object code into assembly language (inverse assembly).

8. EAROM. Electrically eraseable user programmable/ reprogrammable ROM.

9. EEROM. Electrically eraseable (in-circuit) ROM. Similar in concept to a core memory.

10. EPROM. Ultraviolet eraseable user programmable/ reprogrammable ROM.

11. Firmware. See Section 3.1.

12. Firmware Intensive. See Section 3.1.1.

13. Gate. An elemental logic circuit (AND, OR, NOT, etc.).

14. Gate Array. A device for directly implementing logic equations (also FPLA, PAL, PMUX, etc.).

15. Hardware-Intensive. See Section 3.1.1.

16. <u>In-Circuit Emulator</u>. A device for emulating the logical and electrical characteristics of a microprocessor in its native circuit environment (see Appendix B).

17. <u>Large-Scale Integration.</u> Circuit densities of 1000 gates per chip and greater.

18. <u>Logic Analyzer.</u> A device for monitoring and displaying digital logic signals.

19. <u>Memory-Mapped I/O.</u> I/O devices addressed as memory locations.

APPENDIX B

TOOLS AND TECHNIQUES FOR MICROPROCESSOR DEVELOPMENT AND SUPPORT

PURPOSE

Although microprocessor systems are conceptually divided into hardware, firmware, and software subsystems, functionally they operate as a digital system. Implementing a given subsystem in firmware or software rather than hardware buys you increased flexibility but only if appropriate tools exist to capitalize on it.

The purpose of a tool is to

- Provide visibility into system operation.
- Automate repetitive control tasks.
- Collect and reduce data on system behavior.

No automated tool known can <u>prove</u> program correctness. Hence the best tool will provide controlled exercise of the microprocessor with the maximum visibility, speed, and ease of use, and collect and reduce the most behavioral data for ultimate <u>human</u> analysis.

TYPES

Tools unique to microprocessors and firmware are presented below.

- In-Circuit Emulator (ICE) - A device which is substituted in place of a microprocessor and which duplicates its operation both logically and electrically. Usually operated in a master/slave relationship in conjunction with an MDS microprocessor acting as master: plugging the slave microprocessor (ICE) in place of the target microprocessor extends the capabilities of a MDS to those of powerful Computer Monitor and Control (CMAC). A memory-mapping capability allows the target microprocessor to utilize MDS memory as if it were its own. In addition to substituting for the target microprocessor and interfacing to the MDS, the ICE usually contains trace buffers and other diagnostic aids. An ICE may also come as a self-contained system packaged in an attache case for field use. This is a device that is unique to LSI technology and has almost unlimited utility.

- Logic Analyzer - A device for monitoring other digital devices on the logical level. Displays timing information and logic levels. Displays numerical data in a variety of formats, including as disassembled machine instructions.

- Memory Mapping - Substituting one memory for another via a real-time address translation map.

- Microcomputer Development System (MDS) or Microcomputer Prototyping System (MPS) - A microcomputer configured as a stand-alone system for software/firmware development and support. Besides the features of minicomputer systems (mass memory, operating systems, CRT's), it has the capability of interfacing with a PROM Programmer (or other firmware configuration tool), and an In-Circuit Emulator (ICE) or other microprocessor debug tool. The mass memory device may consist of anything from paper tape to disc, usually floppy disc (diskettes). Lack of capability to support multiple users may be a problem.

- Microprocessor Analyzer - Similar in capabilities to an ICE, but clipped onto the microprocessor leads instead of substituted for the microprocessor.

- PROM Programmer - A tool for imprinting (programming) and reading the bit patterns of PROM's. May be operated remotely by an MDS, or manually. Can also be used to verify the contents of PROM'S.

- ROM Simulator - A writeable RAM used to emulate a ROM to enhance firmware debug prior to burn-in. May be purchased separately or as capability in an ICE.

- Signature Analysis - A method for isolating faults to the node level in a logical circuit. Requires additional circuitry built into the system.

- Trace Buffer - A memory for the storage of real-time microcomputer bus signals for use in logic testing. Usually runs under control of an MDS in conjunction with an ICE.

- Universal Development System - An MDS with capability of multiprocessor support, including interface with multiple ICE's.

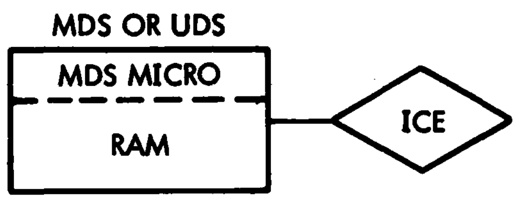
USE

The MDS with an ICE, Logic Analyzer, and a PROM Programmer is currently the most effective combination known for microprocessor life cycle support. The Universal Development System (UDS) has the further attractive capability of supporting, with appropriate adapters, multiple dissimilar microprocessor types. Their use in the life cycle is briefly described as follows[†]

- Development Test and Evaluation Phase

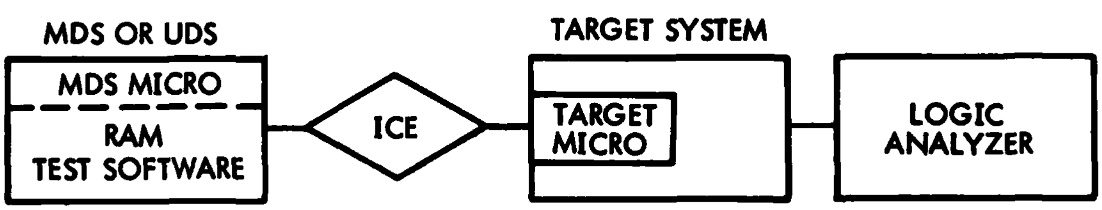
  A. The MDS (or UDS) may be used in advance of hardware development to code and test all internal microprocessor routines.[‡] The use of the ICE allows emulation of the stand-alone microprocessor and symbolic debug of the program. No firmware memories need to be used at this stage, as the MDS RAM will emulate them through the memory-mapping feature.

STAND-ALONE DEVELOPMENT OF SOFTWARE



  B. Effective use may also be made of a MDS/ICE to debug microprocessor hardware by executing software diagnostics and I/O drivers.

DEBUG OF TARGET SYSTEM HARDWARE

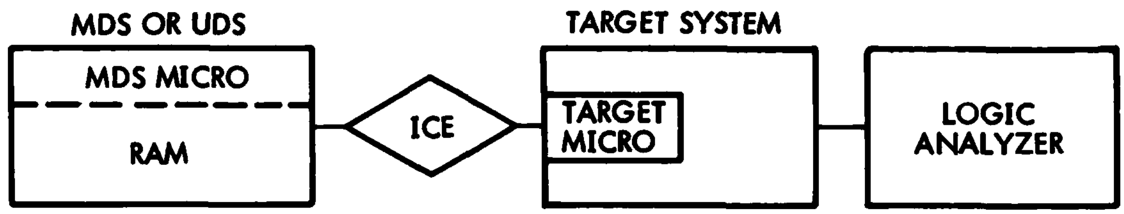


[†] Logic analyzers are useful for hardware debugging throughout this cycle, especially in areas of timing, and parts of the circuit relatively invisible to the microprocessor.

[‡] Cross-compilers and assemblers may also be used at this stage.

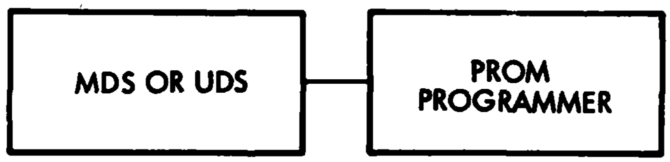
C. Then the software may be exercised under MDS control using the ICE in the final circuit environment of the microprocessor. At this point the MDS RAM (using the memory-mapping feature of the ICE) is still used to emulate the target PROM or ROM.

DEBUG OF TARGET SYSTEM EMULATED FIRMWARE



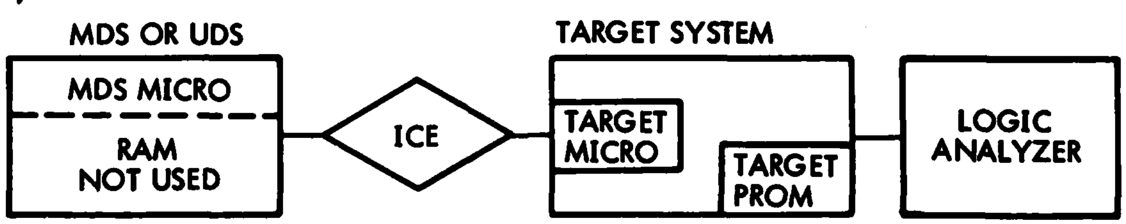
D. Next, the PROM may be imprinted using the PROM Programmer under control of the MDS.

BURN-IN OF FIRMWARE



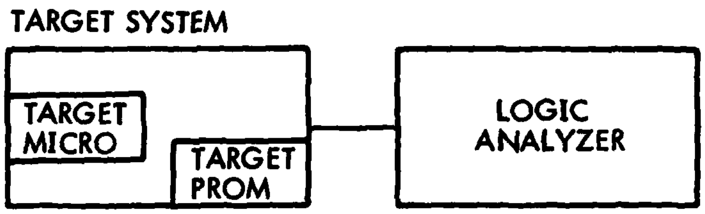
E. Now the PROM's may be plugged into the target system, and the code executed using the ICE under control of the MDS.

DEBUG OF TARGET SYSTEM WITH FIRMWARE IN PLACE



F. Finally, the ICE may be removed, the microprocessor put in its place, and the system operated alone. Here there may still be firmware self-instrumentation and other controls and displays built into the target system.

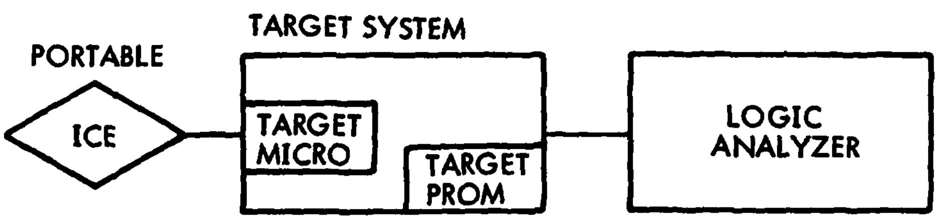
STAND-ALONE OPERATION OF TARGET SYSTEM



- O&M Phase

G. The ICE can be taken to the field to assist in maintenance of the microprocessor. The MDS with the PROM Programmer is used in stand-by mode for possible program patches.

FIELD MAINTENANCE



APPENDIX C

ANNOTATED BIBLIOGRAPHY

SYLVESTER, R.J., and J. GEBELE, White Paper on AFSC Microprocessor Policy, 1 May 1979, ASD/EN, WPAFB OH 45433.

Addresses the same fundamental topics as this guidebook.

- Summarizes existing Air Force policy
- Discusses definitions of microprocessor terminology
- Addresses acquisition issues
  - System specification - source of potential device type proliferation.
  - Environment - devices must be evaluated and slash sheets published.
  - Economics - it is not cost-effective to require full AFR-800-14 documentation for hardware - intensive memories.
  - Visibility - hardware/software intensive decision can be made by CRWG with assistance of an Advisory Team.
  - QA - micros are hard to test and the AF should work with industry on this.
  - HOL's for micros - should be used for prejudicial selection but not as a baseline requirement. Non-approved HOL's are satisfactory for most applications, particularly hardware-intensive.
- Addresses support issue
  - Proliferation - AFSC must work with AFLC to insure availability of parts.
  - Spares - storage life must be ascertained.
  - Cryogenics storage - coordinate with system technology update when storage life exceeded.

- Mask library - supply to a custom IC company when parts are required.
- Parts agreements - establish component supplier intent to supply parts during system life cycle.
- Compatible replacements - hinges on interface standards.

- Firmware
  - Technology - should be carefully selected to minimize life cycle support cost.
  - Identification and labelling - still unresolved issue.
  - Development system - AF should foster competition for a standard development system.
  - Documentation - new DID's required for hardware-intensive applications.
- Standardization - AF needs to encourage
- Makes recommendations
  - Modify AFR 800-14 to
    - Incorporate the hardware/software intensive concept.
    - Require contractors/subcontractors to identify and plan firmware developments to minimize system schedule impacts.
    - Delegate waiver authority to the product division focal points (instead of AFSC/XRF and HQUSAF).
  - Establish an Advisory Team at each product division to advise CRWG's on microprocessor issues and industry interface.
  - Develop new DID's for firmware.

- Initiate activities to
  - Determine suitability of industry-developed microprocessor HOL's for AF applications.
  - Perform research on microprocessor self-tests.
  - Define criteria and standards for design simplicity and testability for firmware.
  - Develop better firmware identification and labelling methods.
  - Develop a standard firmware development system.
  - Develop a set of radiation hardening standards and techniques to meet them.
  - Develop methods to qualify and make available spare parts.

ESTELITA, CAPT. TOM, et al., Engineering and Management Principles of Firmware, Dec. 1977, SM-ALC/MMEC, McClellan AFB, CA 95652.

AFLC white paper recommendations include

- Challenge indiscriminate use of firmware - limit to programs that are stable and exhaustively tested.
- Avoid mask ROM - cost advantage is overshadowed by lack of flexibility.
- Physically organize firmware components for ease of maintenance and logistics - consider using sockets when feasible.
- Leave growth margin in both system and subsystem architecture.
- Include internal identification and checksum - good for configuration control and quality assurance.
- Develop effective tests specific to firmware - e.g., comparators and exercisers.

- Use proven winners - i.e., second-sourced components, unless there is some overwhelming reason not to.
- Be critical of reliability data - conduct independent testing.
- Strive for self-sufficiency - establish backups in case components lose manufacturer's support.
- Document adequately both hardware and software.
- Identify firmware requirements early - optimizes LCC.
- Reprogram at depot - it is cost effective to centralize reprogramming hardware, expertise, and documentation; multi-system support savings possible.
- Automate configuration control - use data base management systems.
- Soften the above requirements for hardware-intensive firmware.

MAJOR RECOMMENDATIONS

- Require total firmware LCC analysis before approving firmware.
- Standardize firmware DID's.
- Include firmware in PMD, PMP, and CRISP.
- Standardize along "winning" firmware lines.

JOHNSON, MAJ. ROBERT W., An AFSC Microprocessor Technology and Management Plan, NAECON 1978, Vol. 2, p. 616.

Survey and Summary of AFSC policy on microprocessors. (Discusses RADC developed AFSC microprocessor Technology and Management Plan).

- Device development and radiation hardening - recommends establishment of an AFSC focal point to coordinate activities.
- Microprocessor system design, development, and support - compounded by proliferation of micros and low quality of most support systems.
    - Software development and support - need for classification of alternative approaches, policies and procedures for standardization, and management guidance.

- Languages - need for guidance on HOL vs LOL selection.
- Test and integration - new approaches needed. Recommends investigation of alternate strategies.
- V&V - current techniques inadequate for micro programs. Recommends additional study.
- Architectures - trend is toward distributed systems, increasing the difficulty of evaluating competing designs against the system requirements and each other. More tools and techniques needed to cope with this problem.
- Standard and modular systems - on-going efforts include development of guidelines and approaches for standardized architectures, bus interface units, computer family, and avionic system executive.

- Microprocessor system acquisition and management
  - Acquisition management, documentation, and maintenance - AFSC and AFLC must work together to formulate improved policy.
  - Education and training - general digital training (see Section 3.3.2.2) is recommended.
- Reliability assurance - RADC is pursuing an intensive program for LSI devices
  - Product evaluation - current and planned program considered sufficient for most commercial LSI for next five years, although still under review.
  - Electrical characterization - current and planned program also considered sufficient.
  - Reliability characterization - timeliness of data is currently the biggest problem and still unsolved.
  - Device specification and standardization - RADC now producing slash sheets on LSI devices. EIA is reviewing candidates for standardization.

BABIAK, CAPT. NICHOLAS J., Impact on Support Requirements Caused by Microprocessors in Avionics Systems, NAECON 1979, Vol. I, p. 128.

Addresses AF microprocessor policy from the AFLC point of view.

- Emphasizes need for joint AFSC and AFLC effort.
- Endorses AFSC definitions of microprocessor terminology and suggests addition of hardware/software-intensive definitions.
- Discusses impact of microprocessors on the CRWG - recommends joint AFSC, AFLC advisory team concept.
- Microprocessor proliferation - one way to reduce is to require basing of acquisition decision on availability of support tools.
- Design changes have far reaching impacts on LCC, which are summarized. Stresses need for basing selection of firmware media on LCC.
- Documentation requirements - must be dealt with on system by system basis based on the support posture and then detailed in the CRISP.
- Firmware labeling and identification - no standard system available. Joint AFSC-AFLC efforts underway to define.
- Spares - recommends second sources, storage of life cycle supply, mask library. Discusses difficulty of estimating size of life cycle supply of reprogrammable parts.
- Testing - current techniques inadequate. Desirable to bring memory location accessibility to edge connectors and be able to disable undesirable circuitry. ATE, self test, and standardized external interfaces recommended.

ZIERNICKI, COL. ROBERT S., "Avionics: The Road Ahead",

Air Force Magazine, July 1979, p. 67.

Key Points

- Focus on standardization of interfaces, not technology.
- In the eventual support of avionics systems, it makes no difference to the AF what vendor supplies the systems.

GENERAL TECHNICAL BACKGROUND:

BLAKESLEE, THOMAS R., Digital Design with Standard MSI and LSI, John Wiley & Sons, 1975.

CARLAN, A., et al., "Testing Embedded Microprocessors" AIAA/NASA/IEEE/ACM Computers in Aerospace Conference, Nov. 1977, p. 119.

HILBURN, JOHN L., and PAUL M. JULICH, Microcomputers and Microprocessors: Hardware, Software and Applications, Prentice Hall, 1976.

KORN, GRANINO A., Microcomputers and Small Digital Computer Systems for Engineers and Scientists, McGraw Hill, 1977.

LEWIS, T.G., and J.W. DOERR, Minicomputers: Structure and Programming, Hayden, 1976.

MACKINTOSH, I. M., "Large-Scale Integration: Intercontinental Aspects", IEEE Spectrum, June 1978, p. 51.

McGLYNN, DANIEL R., Microprocessors: Technology, Architecture, and Applications, John Wiley & Sons, 1976.

OSBORNE, ADAM (& Assoc.), The Value of Micro Power, 1974, General Automation, Inc., Doc. No. 89A00 124A-A.

QUEYSSAC, DANIEL, "Predicting VLSI's Impact on Microprocessors", IEEE Spectrum, May 1979, p. 38.

SCHNEIDER, G.M., "Pascal: An Overview", Computer, April 1979, p. 61.

WEBSTER, JOHN G., and WILLIAM D. SIMPSON, Software for Microprocessors, TI Learning Center, 1976.

*U.S.Government Printing Office: 1980 — 657-084/25